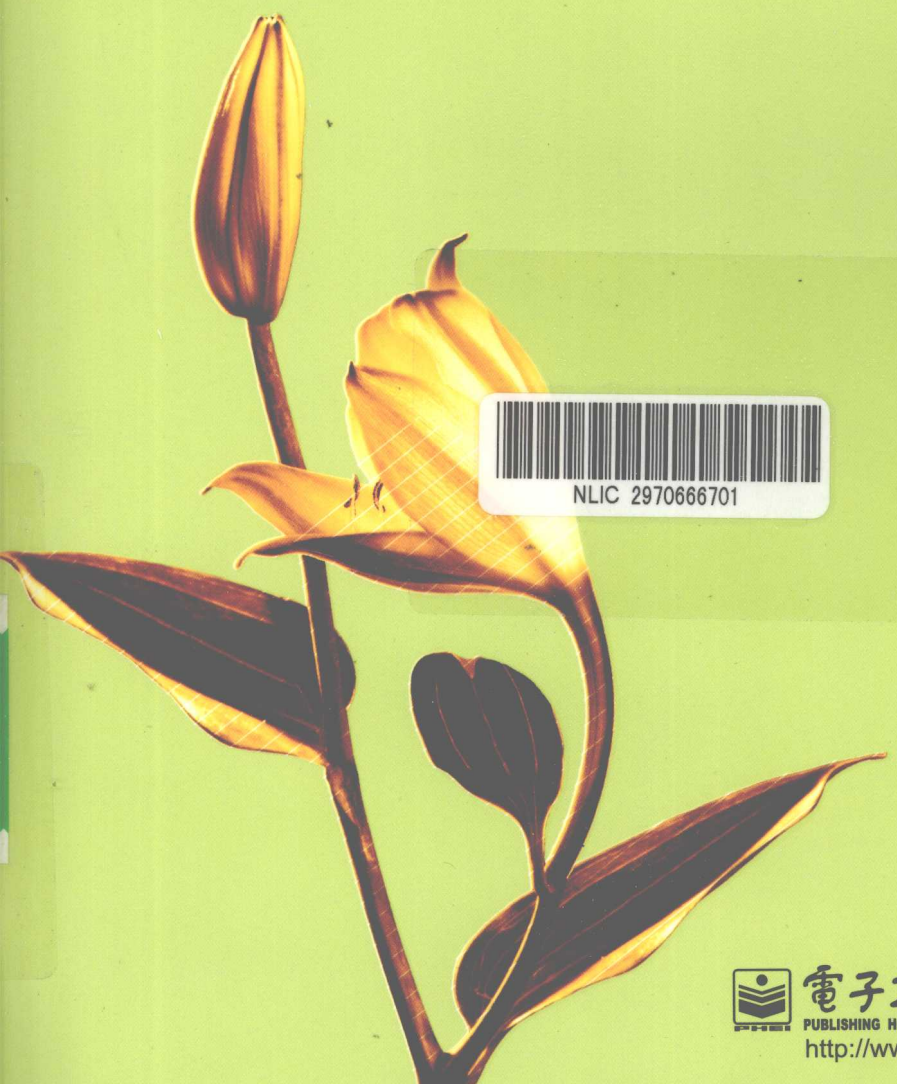
省精品课程教材

C++简明教程

皮德常 编著



高等学校工程创新型『十二五』规划计算机教材

Engineering Innovation

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

高等学校工程创新型“十二五”规划计算机教材

省精品课程教材

C++简明教程

皮德常　编著

電子工業出版社
Publishing House of Electronics Industry
北京 • BEIJING

内 容 简 介

本书是江苏省精品课程“面向对象C++程序设计”的配套教材。全书共9章，包括：C++程序设计基础、文件操作、类的基础、继承、多态、虚函数、对象组合、异常处理、标准模板库STL（主要介绍编程常用的string类、容器类、迭代器及其算法等），以及通过ODBC对数据库编程等。全书结合实例讲解C++的基本概念和方法，力求简洁通俗，循序渐进。书中列举了数百个可直接使用的程序示例源代码，并给出了运行结果；同时配有大量的习题，包括编程题和课程设计题目，并免费提供PPT电子教案及例题源代码。

本书适合作为高等院校各专业面向对象C++程序设计课程教材，也是C++ 编程者的理想自学参考书。

图书在版编目（CIP）数据

C++简明教程 / 皮德常编著. --北京：电子工业出版社，2011.1
高等学校工程创新型“十二五”规划计算机教材
ISBN 978-7-121-12634-5
Ⅰ. ①C… Ⅱ. ①皮… Ⅲ. ①C语言—程序设计—高等学校—教材 Ⅳ. ①TP312

中国版本图书馆CIP数据核字（2010）第250930号

策划编辑：童占梅
责任编辑：童占梅
印　　刷：北京市顺义兴华印刷厂
装　　订：三河市双峰印刷装订有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　邮编　100036
开　　本：787×1 092　1/16　印张：18.25　字数：464千字
印　　次：2011年1月第1次印刷
印　　数：4 000册　　定价：29.80元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

本书是江苏省精品课程“面向对象 C++程序设计”的配套教材。

C++是一种实用的程序设计语言，是高校学生学习程序设计的一门必修专业课程，同时也是程序员广泛使用的编程工具。学好 C++，可以很容易触类旁通地学好其他语言，如 Java 和 C#等。

本书在作者总结多年教学实践经验的基础上编写而成。全书以面向对象程序设计贯穿始终，针对初学者和自学者的特点，结合实例讲解 C++的基本概念和方法，力求用简洁、通俗的语言描述复杂的概念，做到深入浅出、循序渐进。

本书的写作特点

（1）本书主要讲解面向对象程序设计理论和编程方法，它是计算机科学与技术及相关专业学生的编程基础。

（2）在内容安排上，提前讲解文件操作（许多书都是最后讲解）。因为文件是很实用、也是比较难学的一章，这为学生进行课程设计和实验做了铺垫。

（3）本书是作者多年教学经验的结晶。几年来，作者一直从事程序设计方面的教学和科研工作，主讲过程序设计方面的多门课程，为此积累了丰富的教学经验。“从实践到理论，再从理论到实践，循序渐进”是作者教学的心得体会，编写教材也不例外，作者深知学生的薄弱环节和学习特点。

（4）本书的内容与时俱进，讲解了 C++的许多新内容，这是其他教材所不具备的。例如，string 类、体现泛型程序设计思想的 STL，以及基于 STL 的程序设计方法、通过 ODBC 对常规关系数据库编程等，许多教材都不全讲解，甚至不讲解这些内容。作者认为，随着 C++的发展，教材也应当与之同步。本书另辟章节专门讲解这些内容，并结合实例给出了具体的综合应用举例，为读者采用 C++进行课程设计和项目开发打下了坚实的基础。

（5）作业安排从易到难，环环相扣。许多学生学过 C++，却不会编程。作者在教学中认识到了这一点。因此，设计了许多与实际有关的习题，并且它们彼此相关。

（6）课程设计。C++课程往往都有课程设计，为便于教师组织教学和学生对课程设计要求的理解，作者在本书的最后给出了**课程设计的基本要求和文档模板**，为课程设计提供了便利。

（7）讲解力求通俗易懂。编写本书的目的是让读者通过自学或在教师的讲授下，能够运用 C++语言的核心要素，进行面向对象程序设计。因此，本书围绕着“**如何进行 C++编程**”展开。为了便于读者的学习，作者力求该书的语言通俗易懂，将复杂的概念采用浅显的语言讲述，便于读者理解和掌握。

本书的编排特点

（1）每章开始均引出本章要讲解的内容和学习要求。

（2）每章安排的习题都具有很强的操作性，能通过计算机验证。

（3）对书中**重要内容采用黑体标记**，特别重要的内容采用下面加点（着重号）标记。

（4）本书强调程序的可读性。书中程序全部采用统一的设计风格。例如，类名、方法名和变量名的定义做到“望名知义”；语句的末尾或下一句的开头放上左大括号，而右大括号自成一行，并采用缩排格式组织程序代码；此外，对程序中的语句尽可能多地使用注释。希望读者模仿这种程序设计风格。

（5）本书包含大量的程序示例，采用 Visual C++ 2005（Express） 版本给出了运行结果。凡是程序开头带有程序名编号的程序，都是完整的程序，可以直接在计算机上编译运行。

（6）本书采用醒目的标记来显示知识点。这些注意和思考标记穿插在内容中，帮助读者尽快找到重要信息。


注意：值得读者关注的地方，往往是容易混淆的知识点。


思考：提出问题，引导读者思考，以培养读者的思考能力。

课时分配

本课程总课时为 40（理论授课）+ 40（实验）+ 40（课程设计），体现了“精讲多练”的教学思想，具体安排建议如下。

序号	教学内容	学时数		
		合计	讲授	实验
1	第 1 章　C++程序设计基础	9	5	4
2	第 2 章　文件操作	9	5	4
3	第 3 章　类的基础部分	11	5	6
4	第 4 章　类的高级部分	12	6	6
5	第 5 章　继承、多态和虚函数	12	6	6
6	第 6 章　异常处理	3.5	1.5	2
7	第 7 章　模板	7	3	4
8	第 8 章　标准模板库 ST	8	4	4
9	第 9 章　数据库程序设计	8	4	4
10	课程设计	40.5	0.5	40

教学支持

本书的电子教案采用 Power Point 2003 制作，可以在讲课时用多媒体投影演示，这可部分取代板书。教师不仅可以使用本教案，还可以方便地修改和重新组织其中的内容以适应自己的教学需要。使用本教案可以减少教师备课时编写教案的工作量，以及因板书所耗费的时间和精力，从而提高单位课时内的知识含量。

本书向使用本教材的教师免费提供电子教案和程序示例源代码。需要的教师可以直接登录华信教育资源网 http://www.hxedu.com.cn 免费注册下载。

感谢读者选择本书，欢迎您对本书的内容提出批评和修改建议，作者将不胜感激。作者联系方式：dc.pi@nuaa.edu.cn。

皮德常
于南京航空航天大学计算机科学与技术学院

目　录

第 1 章　C++程序设计基础

C++是在 C 语言的基础上扩充而成，以其独特的机制在计算机领域有着广泛的应用。本章主要讲述 C++的基本知识，它是对 C 语言的扩充，因此有着承前启后的作用。

1.1　为什么要学习 C++程序设计

随着计算机软硬件技术的发展，计算机应用规模不断提高，在软件开发语言和工具方面不断推陈出新，新语言、新工具层出不穷。目前，国内许多高校，无论是计算机专业或者是非计算机专业，都在开设 C 语言的基础上，陆续开设了 C++语言，并且将它作为学过 C 语言后的一门专业必修课程。

为了解决程序设计的复杂性，美国贝尔实验室于 1980 年开始研制一种“带类”的 C，到 1983 年才正式命名为 C++。在计算机刚发明时，人们采用打孔机直接进行机器指令程序设计，当程序长度有几百条指令时，采用这种方法就困难了。后来人们设计了用符号表示机器指令的汇编语言，从而能够处理更大更复杂的程序。到了 20 世纪 60 年代，出现了结构化程序设计方法（目前的 C 语言就采用这种方法），这使得人们能够容易编写较为复杂的程序。但是，一旦程序设计达到一定的程度，即使结构化程序设计方法也变得无法控制，其复杂性超出了人的管理限度。例如，一旦 C 程序代码达到 25 000～100 000 行，系统就变得十分复杂，程序员很难控制，而开发 C++语言的目的就是为了解决这个问题，其本质是让程序员理解和管理更大、更复杂的程序。因此，采用支持面向对象的 C++语言是时代发展的需要。

C++语言吸收了 C 和 Simula 67（一个古老的计算机语言）语言的精髓，它具有 C 语言所无法比拟的优越性。C++语言在维持 C 语言原来特长（如效率高和程序灵活）的基础上，借鉴了 Simula 67 语言的面向对象的思想，将这两种程序设计语言的优点相结合。C++语言的程序结构清晰、易于扩展、易于维护，同时又不失效率。目前，C++语言已超出了当初设计它的目的，成功地应用到数据库系统、数据通信系统等领域，并成功地构造了许多高性能的系统软件。C++与 C 语言相比，具有三个重要特征，从而使其优越于 C 语言。

（1）支持抽象数据类型 ADT（Abstract Data Type）。在 C++语言中，ADT 表现为类，它是对象的抽象，而对象是数据和操作该数据代码的封装体，它提供对代码和数据的有效保护，可防止程序其他不相关的部分偶然或错误地使用对象的私有部分，这是 C 语言所无法实现的。

（2）多态性。即一个接口，多重算法。C++语言既支持早期联编又支持滞后联编，而 C 语言仅支持前者。

（3）继承性。一方面它保证了代码复用，确保了软件的质量；另一方面也支持分类的概念，从而使对象成为一般情况下的具体实例。

以上三个特性，我们将在后面的章节详细讲解。

目前许多系统软件，如操作系统、数据库管理系统 DBMS 等都采用 C++语言编写，所以从事软件开发和计算机应用的人员，不掌握 C++语言简直寸步难行。一句话：掌握 C++编程已成为各专业学生的必然选择。

1.2 过程化程序设计和面向对象程序设计

C++语言支持过程化程序设计和面向对象程序设计，它们是两种不同的编程模式。

在过程化程序设计中，程序员编写函数（有些书称为过程）。这些函数是执行某个特定任务的程序语句的集合，每个函数一般包含局部变量和全局变量。过程化程序设计是以过程（函数）为核心，而面向对象程序设计（Object Oriented Programming，OOP）是以对象为核心。一个对象是一个包含数据和对数据操作的封装体。对象包含信息和对信息操作的能力，并且对信息的操作基于消息传递。

随着我们对 C++语言学习的逐步深入，读者将会深刻理解过程化程序设计和面向对象程序设计的不同。

1.3 简单的输出和输入方法

C++语言除了具有 C 语言的输出和输入方法外，还具有自己的输出和输入方法。简单的输出是通过 cout 对象实现，简单的输入是通过 cin 对象实现，下面分别讲述它们的用法。

1.3.1 cout 对象

cout 对象只能用来输出数据，同时也称为标准输出对象，它的作用是使用标准输出设备（即显示器）输出信息。

注意：cout 可以看作是 console output 英文单词的缩写。

cout 是输出流中的一个对象，因此也称为**流对象**。它的操作对象是数据流，要输出信息，只需将数据流传给 cout，例如：

```
cout << " I like programming language C++" ;
```

在上述语句中，“<<”是**流插入操作符**，它的功能是将字符串" I like programming language C++"送给 cout。

【例 1-1】数据输出方法示例。

```
#include <iostream>
using namespace std;
int main( )
{
    cout << " I like programming language " << " C++" ;
    return 0;
}
```

【程序运行结果】

```
I like programming language C++
```

注意：程序的开头必须包含 iostream 文件，因为 cout 对象（和下一节的 cin 对象）就定义在该文件中，这就像 C 程序必须包含 stdio.h 文件一样。

通过上例可以看出，使用<<可以传送多个数据给 cout。例如，将例 1-1 的 cout 语句修改

如下：

```
cout << " I like programming language " ;
cout << " C++" ;
```

程序段的运行结果和例 1-1 相同。但我们要理解一个重要的概念：虽然输出分解为两个语句，但程序仍在同一行上显示信息。除非你指定输出方式，否则送给 cout 的信息将会连续显示。

```
cout << "我最喜爱的东西: " ;
cout << "computer " ;
cout << " & tea " ;
```

程序段的运行结果：

```
我最喜爱的东西: computer  & tea
```

输出结果与源代码中字符串的安排是不同的，cout 完全按照提交数据的方式输出。在上面的源代码中使用了三个 cout 输出语句，但它仍在一行上输出信息，这是因为如果不加入换行符，cout 不会自动换行。**有两个换行的方法，一个是在 cout 语句后加一个流操作符 endl；另一个是加入'\n'换行符。**

【例 1-2】换行输出示例。

```
#include  <iostream>
using namespace std;
int  main( )
{
    cout << "我最喜爱的东西: "<< endl ;
    cout << "computer "<< '\n' ;
    cout << " & tea \n" ;
    return 0;
}
```

【程序运行结果】

```
我最喜爱的东西:
computer
 & tea
```

注意：endl 是 end of line 的缩写，它和'\n'的功能一样，都是换行。当 cout 遇到'\n'时，将输出光标移到下一行的开头。此外，不要把反斜线 \ 和正斜线 / 弄混，'/n'是不会换行的；同时也不要在反斜线和字符 n 之间加空格，如'\ n'是错误的。

1.3.2 cin 对象

cout 是 C++语言的标准输出对象，cin 是 C++语言的标准输入对象，它的功能是从 I/O 控制台（即键盘）接收数据输入。

【例 1-3】 标准输入和输出示例。

```
#include  <iostream>
using namespace std;
```

```
int  main( )
{
    int  length, width, area ;

    cout << "计算矩形的面积 \n" ;
    cout << "输入矩形的长: " ;
    cin >> length ;
    cout << "输入矩形的宽: " ;
    cin >> width ;
    area = length * width ;
    cout << "矩形的面积为: " << area << "\n" ;
    return 0;
}
```

【程序运行结果】

```
计算矩形的面积
输入矩形的长: 10 [Enter]
输入矩形的宽: 20 [Enter]
矩形的面积为: 200
```

【程序解析】这个程序用来计算一个矩形的面积。当程序运行时，用户输入的数据将存储在 length 和 width 两个变量中，如下两行：

```
cout << "输入矩形的长: " ;
cin >> length ;
```

cout 在屏幕上显示“输入矩形的长:”，cin 为 length 变量输入值。其中“>>”称为**流提取操作符**，它从左边输入流对象 cin 中读一个数，并把它存储在>>右边的变量中。在上面的程序语句中，cin 将从键盘输入的数据存储在 length 变量中。

注意：插入操作符“>>”和提取操作符“<<”指定了数据的流动方向。插入操作符“>>”将输入的数据传给变量，而提取操作符“<<”将变量（或常量）的值传给 cout 输出。cin 对象在读取数据时，将暂停程序的运行，直到从键盘上输入数据并按 Enter 键确认。cin 对象能自动地将输入数据转换成与变量一致的数据类型。例如，用户输入“10”，cin 将分别读入字符 1 和 0，在将该数据存储到变量之前，cin 能够自动地将它们转换成整数 10。cin 同样能够识别出，像“10.7”这样的数不能存储在整型变量中。如果用户输入一个浮点数给整型变量，那么小数点后的位数将被舍弃(也称截断)；如果用户输入浮点数，cin 通过截断浮点数后的小数部分，将整数部分存储在整型变量中。

如果程序要求输入数据，那么就应该向用户提示该输入什么样的数据。例 1-4 由于缺乏提示用户的信息，不是一个好程序，我们在写程序中要杜绝这种现象。

【例 1-4】缺乏输入提示信息的写法，这是一种不好的写法。

```
#include  <iostream>
using namespace std;
int  main( )
{
```

```
    int  length, width, area ;

    cin >> length ;
    cin >> width ;
    area = length * width ;
    cout << "矩形的面积为: " << area << "\n" ;
    return 0;
}
```

【程序解析】当运行这个程序时，用户面对的是黑屏幕，不知道要做什么事。一个功能完善的程序应当及时、友好地给用户必要的提示信息。

采用 cin 对象可以一次读入多个变量的值，例如：

```
int  length, width, area ;
cout << "请输入矩形的长和宽，中间用空格隔开: " ;
cin >> length >> width ;      // 给两个变量读取值
```

下列语句等待用户输入两个数值，并把输入的第一个数赋给变量 length，第二个数赋给变量 width：

```
cin >> length >> width ;
```

在上例输出中，用户输入 10 和 20，10 就送给了变量 length，20 送给了变量 width。当输入多个数值时，数值之间要加空格。cin 读到空格时，就能够区别输入的各个数值了，数值间的空格数无所谓，如用户按照如下形式输入也可以：

```
10          20
```

但要注意，在最后一个数输入后要按[Enter]键。

 **注意**：cin 语句读取数据的特点和 C 语言中的 scanf 函数类似，在上例的输入中，如果先输入 10 并按[Enter]键，然后输入 20 再按[Enter]键，完全可以正确地输入数据。本书对数据的输入和输出格式不做过于详细的探讨，因为这些不是 C++的核心和重点。

采用一个 cin 语句，也可以同时为多个不同类型的变量读入数据。

【例 1-5】采用 cin 语句同时为多个不同类型的变量输入数据。

```
#include  <iostream>
using namespace std;
int  main( )
{
    int   whole ;
    float  fractional ;
    char  letter ;

    cout << "请输入一个整数、一个浮点数和一个字符: " ;
    cin >> whole >> fractional >> letter ;
    cout << "整数: "<< whole << endl ;
    cout << "浮点数: "<< fractional << endl ;
    cout << "字符: "<< letter << endl ;
```

```
    return 0;
}
```

【程序运行结果】

```
请输入一个整数、一个浮点数和一个字符：100   3.14159   Y [Enter]
整数：100
浮点数：3.14159
字符：Y
```

【程序解析】从上例的输出可以看出，各个数值分别储存在各自的变量中。如果用户的输入如下：

```
5.7   4   B
```

那么，程序将把“5”存储在变量 whole 中，将“0.7”存储在变量 fractional 中，把“4”存储在变量 letter 中，所以我们必须以正确的格式输入各个数值。

cin 语句读入字符串的方式与 scanf 函数类似，也是采用字符数组存储字符串，例如：

```
char  company[12] ;
cin >> company ;
```

该数组最多能存储 12 个字符，并且最后一位应是'\0'，表示字符串的结束。

注意：如果用一个字符数组存储字符串，要确保该字符数组足够大，能够存储字符串中的所有字符（包括空字符'\0'）。

【例 1-6】采用 cin 语句读取一个字符串。

```
#include  <iostream>
using namespace std;
int  main( )
{
    char  name [21] ;

    cout << "What is your name? " ;
    cin >> name ;
    cout << "Hi" << name << endl ;
    return 0;
}
```

【程序运行结果】

```
What is your name? Zhang san [Enter]
HiZhang san
```


注意：cin 对象允许用户输入一个比字符数组容量大的字符串，但字符串会溢出，从而破坏内存中的其他数据，因此在输入字符串时，不要超出字符数组的有效容量。

1.4 标识符

标识符是由程序员定义的记号，用来标识程序中元素的名字。变量名是标识符的一种，

在程序中，只要不使用 C++预定义的关键字，你可以随意选择变量名。关键字是 C++的“核心”，它有特定的用途。表 1-1 给出了 C++关键字，它们全都是小写字母。

表 1-1　C++关键字表

asm	auto	break	bool	case
catch	char	class	const	const_cast
continue	default	delete	do	double
dynamic_cast	else	enum	explicit	extern
false	float	for	friend	goto
if	inline	int	long	mutable
namespace	new	operator	private	protected
public	register	reinterpret_cast	return	short
signed	sizeof	static	static cast	struct
switch	template	this	throw	true
try	typedef	typeid	typename	union
unsigned	using	virtual	void	volatile
wchar_t	while			

定义标识符应尽量选择有含义的英文单词，下面的变量命名就毫无意义：

```
int  x ;
```

因为我们从“x”看不出变量的任何含义，而下面的命名就比较好：

```
int  itemsOrdered ;
```

通过变量名 itemsOrdered，可以很容易地看出该变量的意义。这种编程风格使程序易于理解和维护，因为一个大的系统通常由数万行源代码构成，程序的可读性十分重要。

标识符是区分大小写的，变量 itemsOrdered 和 itemsordered 是不同的变量。我们之所以将变量 itemsOrdered 中的“O”大写，是为了增强程序的可读性，例如：

```
int  itemsordered ;
```

是清一色的小写字母，使人不易读懂，因此这不是好的变量名。

有些程序员喜欢使用下划线连接单词，构成变量名，这个风格也不错，例如：

```
int  items_ordered ;
```

总之，选择标识符时，**要尽可能做到“见名知意”，选择有含义的单词符号作为标识符，使别人（包括你本人）容易读懂程序。**

1.5　布尔类型

为了纪念英国的数学家乔治·布尔（George Boolean），在程序设计语言中引入了“布尔”类型。我们在 C 语言中，已经学习过了布尔表达式，它是一个具有真值或假值的表达式。C 语言并没有提供布尔数据类型，C++语言为了方便程序员的使用，引入了布尔类型。

在布尔数据类型中，**将非 0 值解释为 true，将 0 值解释为 false。**

【例 1-7】布尔型变量的输入和输出。

```
#include  <iostream>
using namespace std;
```

```
int  main( )
{
    bool  bValue ;

    bValue = true ;     // true 实际上就是数值 1
    cout << bValue << "    " ;
    bValue = false ;   // false 实际上就是数值 0
    cout << bValue << endl ;
    return 0;
}
```

【程序运行结果】

```
1    0
```

【程序解析】从程序的输出可以看出，true 是用 1 表示，false 是用 0 表示。布尔型变量虽然不常用，但它对条件表达式的判断非常有用。

 注意：从程序程序设计语言原理的角度讲，布尔型变量的取值只能是真或假，但 C++ 语言中对它的定义不够严格，将 0 看作 false，将非 0 看做 true。这是为了向下兼容，即兼容它的子集 C 语言，因为 C 语言就是这样定义的。

1.6 培养良好的编程风格

程序员使用标识符、空格、Tab 键、空行、标点符号、代码缩进排列和注释等，来安排源代码的方式，就构成了编程风格中重要的组成部分（此外还有程序设计方法等）。

1.6.1 风格对比

当编译器对源程序进行编译时，它会将程序处理为一个长字符串。一条语句不在同一行或操作符与操作数之间有空格，都不会影响编译。但阅读程序的人很难读懂这种书写不规范的程序。例 1-8 虽然没有什么语法错误，但却很难读懂。

【例 1-8】糟糕的编程风格。

```
#include  <iostream>
using namespace std;
int  main(){float  shares=220.0 ;    float  avgPrice=10.67f; cout
<<"There were "<<shares<<" shares sold at RMB"<<avgPrice<<"per share.\n" ; return
0;}
```

上面的程序虽然没有违反 C++语法规则，但却难以阅读。理想的编程风格是，在编程时使用空格和代码缩进排列，从而使别人能很快读懂你的程序。

 注意：尽管编程风格的自由度很大，但我们还是应该遵循程序设计的国际常规。这样，其他程序员才会很容易读懂你的程序。本书的编程风格是国际编程风格中的一种，我们建议初学者模仿我们的编程风格。当然，你也可以不模仿我们的风格，而参考国际知名公司的编

程风格，如微软、IBM、Borland 和 SUN 公司等，他们的风格都很好。不同的公司风格也不尽相同，可以参考他们提供的样式文件，如头文件，这样对培养你的编程风格大有帮助。

除了前面提到的编程风格以外，还要注意如何处理一行中太长的语句，C++是流式语言，它允许将长语句写在不同行。例如，将一个 cout 语句用五行书写：

```
cout << "华氏温度为："
     << fahrenheit
     <<"摄氏温度为："
     << centigrade
     << endl ;
```

当然，也可以将上面的 cout 语句写在一行上。变量的定义也可以分行写，例如：

```
int  fahrenheit,        // 存储华氏温度
     centigrade ;       // 存储摄氏温度
```

总之，如果你的编程风格不够好，就请模仿我们的风格吧！

1.6.2 注释方法

注释是为程序员提供的，用来说明程序或解释代码的某些方面。注释是程序的组成部分，但编译器在编译时会忽略它，不构成可执行代码。它也属于编程风格中关键的一环。

许多程序员都会在源代码中尽可能少地加注释语句，因为编写源代码本身已经够痛苦了。但是，养成加注释的习惯是会有帮助的，虽然编程时可能要花费额外的时间，但在以后纠错和调试程序时会节省大量的时间。假设你辛苦数月编写了一个大约 8 000 到 10 000 行的 C++程序代码，完成了编码，并且调试成功，交给客户使用，你也继续做下一个项目。但一年以后需要对程序进行修正，当你打开数万行没有注释的源代码时发现，其中的许多函数你已经不知道它们的功能了，当初设计的目的是什么都不知道了。这时你就会想，要是当初加一些必要的注释就好了，但为时已晚，你目前要做的只能是用较多的时间来理解源程序，或完全重写。

上述假设可能有些极端，只是现在还没有发生在你身上。现实程序往往规模大、构造复杂，必须给程序添加必要的注释，增加程序的可读性，使程序易于维护。

C++语言支持两种注释方法。

一种方法是以双斜线“//”开始，直到本行结束，它可以出现在程序的任意一个地方，例如：

```
//*********************************************************************
// 作        者：张  三
// 功        能：计算 XXX 公司的员工的工资
// 最后修改时间：2010 年 12 月 3 日
//*********************************************************************
#include  <iostream>
using namespace std;
int  main( )
{
   float  payRate ;       // 存储单位小时内的工资
   float  hours ;         // 存储员工已经工作的小时数
```